## Слайд “Цели работы”

Добрый день. Целью моей работы является сравнение нескольких рендеров, те программ для отрисовки изображений, и алгоритмов трассировки, реализованных в них. Проведя эксперименты мною были сделаны выводы о скорости работы этих алгоритмов и о реалистичности изображений, полученных в результате их применения.

*Чтобы* провести корректное сравнение мне, во-первых, необходимо было смоделировать различные сложные для просчета сцены. В них должны были быть различные сложноустроенные модели освещения и множество вариаций различных поверхностей.
После настройки рендерился набор из нескольких изображений для сравнения за некоторое конечное время и эталонное изображение, которое рендерилось за условно бесконечное время.
*Дальше* проводилось сравнение с помощью функции psnr (отношение пикового уровня сигнала к шуму).

## Слайд “Реалистичный рендеринг”

### Актуальность работы.

Сейчас существует много программ трехмерной визуализации, которые применяются в кино- и телеиндустрии, дизайне, архитектуре и т.п., Они используют для реалистичного синтеза различные алгоритмы трассировки лучей. Сегодня наибольшей популярностью пользуются платные пакеты, такие как Maxwell Render, Lumion, Solidworks Visualize, Marmoset Toolbag, Corona Render, Iray, V-Ray, Indigo Render. Однако для независимых разработчиков и некоторых молодых компаний значительный расход средств на их приобретение может быть критическим, поэтому они склонны искать бесплатные аналоги. К счастью, такие существуют, но тогда встает вопрос о качестве их работы, ведь помимо цены, важнейшими характеристиками являются время рендеринга и фотореалистичность получающегося изображения. Также важный фактор, влияющий на время – это реализация распараллеливания алгоритмов рендеринга. Поэтому также актуально рассмотреть и реализацию многопоточности у этих алгоритмов.

*На данных* изображениях можно видеть результат работы каждого из исследуемых рендеров.

На первом рисунке отрисованы различные поверхности, такие как стекло, пластик, древесина, металл и тд.
На втором рисунке используется технология подповерхностного рассеивания для реалистичного отображения кожи.

На третьем можно увидеть поверхность воды, учитывающую закон Френеля, и множество источников освещения.
**Важное замечание по поводу реалистичности**: Данные изображения созданы профессионалами и приведены здесь для понимания возможностей, которых можно добиться используя их. Я не являюсь профессионалом, поэтому реалистичность полученных мною изображений является условной.

не читать
!**Таким образом**, *актуальность* заключается в том, что в работе сравниваются эти характеристики у некоторых популярных бесплатных рендеров и исследуется их оптимальная применимость к визуализации различных световых явлений.!
**Но как все это работает?**

## Слайд “Концепция PBR”
Реалистичный рендеринг базируется на 2 вещах: алгоритмы трассировки и уравнение рендеринга.  Уравнение рендеринга это интеграл, который просчитывает энергетическую яркость поверхности с учетом собственной энергетической яркости и энергетической яркости падающего под любым углом светового луча.
!не читать!L_e в данном уравнении это собственная энергетическая яркость поверхности, те грубо говоря, собственное излучение света. L_i это энергетическая яркость падающего луча из направления i. это направление определяется вектором “омега малое и”. Омега большое, это полусфера, которая учитывает любое направление i !
f_r это двухлучевая функция распределения рассеивания. Эта функция отвечает за то, как определенная точка поверхности реагирует на падающий свет. На рисунке 4 вы можете видеть примеры различных поверхностей, полученных с помощью различных вариаций этой функции.

## Слайд “Алгоритмы трассировки”

Рассмотрим теперь вторую составляющую рендеринга.
Цифровое изображение представляется в виде набора пикселей. Для того чтобы вычислить дискретные значения пикселей, необходимо взять за образец исходную непрерывно определяемую функцию изображения, которой является уравнение рендеринга. Единственный способ получить информацию о функции изображения – это взять ее образец с помощью трассировки лучей.
Изображение может быть создано только путем выборки значений функции в конкретных позициях пикселей. Чем больше выборок для разных пикселей, тем лучше качество.

**Первый алгоритм** - самый простой но и самый ресурсоемкий. Он был описан в той же статье 86 года, где впервые было описано уравнение рендеринга. Этот алгоритм заключается в симулировании пути света начиная от наблюдателя. На изображении это глаз. Попадая в любую точку поверхности, как можно видеть на правом изображении, каждый первичный луч, который обозначен большой стрелкой, формирует вторичные лучи, отскакивающие под разным углом и уходящие в пространство сцены.
!Метод Монте-Карло (MC) отличается от метода трассировки лучей количеством первичных лучей: там, где трассировка лучей подразумевает группу из нескольких первичных лучей, Монте-Карло использует только один первичный луч. При отскоке от поверхности первичный луч разбивается на группу из вторичных лучей. !
Вторичные лучи, в свою очередь дойдя до поверхностей объектов поглощаются. И если точка, в которой поглотился вторичный луч не была освещена источником света в сцене (например лампой), то энергетическая яркость света в этой точке будет равна 0.
!В каждой точке падения вычисляется, так называемая **оценка монте-карло** - числовой ряд, который приближенно вычисляет значение уравнения рендеринга. !
Отражения вторичных происходят до полного поглощения луча, которое в данном алгоритме определяется по степени вклада в конечный результат. Они реализуются с помощью рекурсии, которая останавливается, если энергетическая яркость близка или равна нулю.

Второй алгоритм - Метод фотонных карт. Он состоит из двух этапов.

**На первом** этапе световые лучи, которые содержат некоторую дискретную информацию, выпускаются из источников освещения, как жто видно на левой картинке, и для каждой точки на поверхности сохраняется информация об ее освещенности в кешэ.
**На втором этапе** лучи выпускаются уже из камеры и при попадании на поверхность информация об освещении собирается в некоторой окрестности точки падения, которая изображена в виде пунктирной окружности.
**Этот алгоритм содержит проблему кэша**, когда кол-во информации об освещении может быть ограничено количеством памяти. В рендерах реализован модифицированный  алгоритм, который называется **стохастический прогрессивный метод фотонных карт**, в котором, во-первых, пер**вый** и вто**рой** этапы меняются местами, а во вторых используется более ресурсо экономичная модель.

Третий алгоритм - двунаправленная трассировка пути, в котором строится сначала путь из камеры, на картинке x1, x2 , потом строиться путь из источника света y1, y2 и потом они соединяются.

## Слайд “Сцены для сравнения рендеров”

На данных изображениях представлены эталонные сцены, с которыми сравнивались полученные наборы.

Первый эксперимент - модель стакана с стеклянной поверхностью, в основном исследуется способность рендеров и алгоритмов вычислять каустики.

Во втором эксперименте достаточно простая модель, однако в некоторых местах здесь достаточно сложные пути света из-за преломлений через стеклянные поверхности бутылок и отражений от зеркала.

В третьем эксперименте исследуется способность симулирования пропускающей свет ткани. В верхнем проеме окна стоит стекло, в нижнем ничего нет.

В четвертом эксперименте исследуется способность рендеров реалистично симулировать поверхность жидкости и всевозможные световые явления, проявляющиеся при этом. В данной сцене существует два источника света, которые не совсем источники, а материал с собственной яркостью, те тут как раз используется компонент L_e. Свет проходит через стеклянную поверхность и поверхность воды.

Пятый эксперимент является наиболее сложным для алгоритмов трассировки. это модель световода с двумя прямыми углами. Из-за этого большая часть световых лучей отсеивается. Индекс преломления стекла световода = 1.8.

## Слайд “Результаты”

Сегодня я расскажу о результатах первых двух экспериментов.

В первом эксперименте

0. **В appleseed**

Используется алгоритм SPPM
Для расчета прямого освещения используется SPPM
Кол-во испускаемых лучей из источника: 10M;
Радиус окрестности: 0.1;

Начальное кол-во отскоков возможных отражений до ограничения семплирования с помощью русской рулетки: 6
Кол-во итераций (семплов): 25, 52, 156, 370.

1. **Cycles**

Используется алгоритм Path Tracing.
Максимальное кол-во отражений/преломлений лучей испускаемых от камеры от:

Диффузной поверхности – 780;
Глянцевой поверхности – 9750;
Пропускающей поверхности – 7800;

Кол-во итераций (семплов): 17, 32, 89, 195.

2. **В LuxCore**

Используется алгоритм Path Tracing + Caustic light cache;
Это технология похожа на первый этап метода фотонных карт.
Кол-во испускаемых лучей из источника: 10M;
Радиус окрестности: 0.01;
Максимальное кол-во отражений фотона, выпущенного из источника света: 16;
Максимальное кол-во отражений лучей, испускаемых от камеры от:

Диффузной поверхности – 16;
Глянцевой – 36;
Зеркальной – 38;

Кол-во итераций (семплов): 148, 293, 854, 1898.

В первых трех графиках отражено процентное соотношение увеличения скорости рендеринга при увеличении кол-ва потоков вдвое: с 6ти до 12ти. Можно заметить что все рендеры увеличивают свою скорость примерно на 70%, кроме cycles, у которого 60%.
На рисунках 4-6 отображено соотношения схожести с эталонным изображением. Как видно appleseed и luxcore практически одинаково отображают первую модель. Это связано с тем, что в них используются похожие алгоритмы. В cycles же используется трассировка пути, которой сложно даются вычисления каустик.

## Во втором эксперименте

во всех алгоритмах используется трассировка пути.
1. appleseed

Используется трассировка пути.

Максимальное кол-во отражений/преломлений лучей испускаемых от камеры от:

Диффузной поверхности – 8;
Глянцевой поверхности – 8;

Пропускающей поверхности – 8;
Кол-во итераций 6 потоков (семплов): 7, 14, 27, 54.

2. cycles

Используется трассировка пути.
Максимальное кол-во отражений/преломлений лучей испускаемых от камеры от:
Диффузной поверхности – 16;
Глянцевой поверхности – 6;
Пропускающей поверхности – 8;
Кол-во итераций 6 потоков (семплов): 36, 68, 72, 139.

3. luxcore

Используется трассировка пути.
Диффузной поверхности – 4;
Глянцевой(отражающей) поверхности – 12;
Пропускающей поверхности – 12;
Кол-во итераций 6 потоков (семплов): 88, 173, 335, 676.

Из данного эксперимента касательно увеличения скорости можно сделать аналогичный с прошлым экспериментом вывод - среднее увеличение скорости на 60 %.
Из данной таблицы №4 видно, что appleseed показывает наибольшую схожесть, но следует учесть, что на такой результат влияют:

1) первое, то что полностью идентичный результат в принципе невозможно получить разными рендерами
2) второе, мои не идеальные навыки
3) и третье, в качестве эталонного изображения взят результат модели того же appleseed, но за неограниченное время.

Видно, что наиболее качественный результат получают рендеры cycles и appleseed. Но здесь также важно обратить внимание на скорость роста графиков. Как можно видеть, наименьшая скорость роста у рендера luxcore. Это говорит о том, что этим рендером достигнут пик качества, то есть когда последующие итерации не внесли больших изменений в результирующее изображение. Аналогичный результат показывает и рендер cycles. Отсюда можно сделать вывод, что в данной модели оптимальнее выбирать cycles.

## Слайд выводы

По итогам экспериментов наилучшую динамику показал рендер luxcore, однако наиболее быстрым является рендер cycles. Рендеры appleseed и Luxcore в большинстве случаев демонстрировали схожие результаты: они дали изображения приблизительно равного высокого качества.